Wireless Personal Communications

Secure Data Aggregation In Wireless Sensor Networks To Enhance Fault-Tolerance Using Quadratic Minimum Spanning Tree Approach

--Manuscript Draft--

Manuscript Number:	WIRE-D-16-01127
Full Title:	Secure Data Aggregation In Wireless Sensor Networks To Enhance Fault-Tolerance Using Quadratic Minimum Spanning Tree Approach
Article Type:	Manuscript
Keywords:	Sensor Networks, Q-MST, Fault tolerance, Data aggregation, Data integrity
Corresponding Author:	Priyanka Nehra, M.Tech Central University of Rajasthan AJMER, rajasthan INDIA
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Central University of Rajasthan
Corresponding Author's Secondary Institution:	
First Author:	Aitha Nagaraju, Ph.D
First Author Secondary Information:	
Order of Authors:	Aitha Nagaraju, Ph.D
	Priyanka Nehra, M.Tech(Ph.D)
Order of Authors Secondary Information:	
Funding Information:	DeitY (PhD-MLA/4(40)/2015-16) — Dr. Aitha Nagaraju
Abstract:	In Wireless Sensor Networks (WSN), Secure Data Aggregation is a challenging issue in the fault tolerance situation along with minimum consumption of energy. In the literature Minimum Spanning Tree is a mechanism which is most widely used tree mechanism for data aggregation in WSN. In WSN, nodes are typically densely deployed in disaster environments. The fault tolerant data transmission then becomes a crucial task in Wireless Sensor Network Environment. WSNs are subject to node failure due to many reasons like battery power, physical damage to the sensors, etc. In this proposed work we have compared the performance of Q-MST with Minimum spanning tree (MST) using Prims algorithm for evaluating the Data Aggregation. This work mainly concentrates on the replacement of alternate edge in place of weak edge (resource point of view) in MST with Q-MST which is an Artificial Bee Colony Algorithm (ABC). The proposed work shows the advantage of alternate path instead of regenerating the MST by considering two hop data verification mechanism for data integrity which is SEDAN (Secure and Efficient Data Aggregation Protocol for Wireless Sensor Networks) without interacting with base station for verification of data every time. We have performed simulation using MATLAB Software and we compared the results with the proposed approach Q-MST with MST. We have compared, the proposed work would perform better in the fault tolerance situation.

Powered by Editorial Manager® and ProduXion Manager® from Aries Systems Corporation

Manuscript
Click here to download Manuscript template1.pdf
Click here to view linked References

Noname manuscript No.
(will be inserted by the editor)

Secure Data Aggregation In Wireless Sensor Networks To Enhance Fault-Tolerance Using Quadratic Minimum Spanning Tree Approach

Priyanka Nehra *, Nagaraju Aitha

Received: date / Accepted: date

Abstract In Wireless Sensor Networks (WSN), Secure Data Aggregation is a challenging issue in the fault tolerance situation along with minimum consumption of energy. In the literature Minimum Spanning Tree is a mechanism which is most widely used tree mechanism for data aggregation in WSN. In WSN, nodes are typically densely deployed in disaster environments. The fault tolerant data transmission then becomes a crucial task in Wireless Sensor Network Environment. WSNs are subject to node failure due to many reasons like battery power, physical damage to the sensors, etc. In this proposed work we have compared the performance of Q-MST with Minimum spanning tree (MST) using Prims algorithm for evaluating the Data Aggregation. This work mainly concentrates on the replacement of alternate edge in place of weak edge (resource point of view) in MST with Q-MST which is an Artificial Bee Colony Algorithm (ABC). The proposed work shows the advantage of alternate path instead of regenerating the MST by considering two hop data verification mechanism for data integrity which is SEDAN (Secure and Efficient Data Aggregation Protocol for Wireless Sensor Networks) without interacting with base station for verification of data every time. We have performed simulation using MATLAB Software and we compared the results with the proposed approach Q-MST with MST. We have compared, the proposed work would perform better in the fault tolerance situation.

Keywords Sensor Networks, Q-MST, Fault tolerance, Data aggregation, Data integrity

---

F. Author
Central University of Rajasthan, Ajmer, India
Tel.: +91-9414766758
E-mail: priyanka_nehra@yahoo.com

S. Author
Central University of Rajasthan, Ajmer, India

1 Introduction

A WSN is composed of a large number of small sized sensor nodes that have low computation power, low battery energy, fixed radio range, self-organizing capabilities etc. They sense the data from the physical environment, perform some computation on the sensed data and transmit it either to intermediate node or to the sink node. The three basic jobs of a sensor node are sensing, processing and transmitting the data [1] [3].

WSN used in many applications as In military for tracking enemies, tracking animals, traffic tracking, for monitoring temperature, humidity, pressure, inventory monitoring, forest fire detection and many more [2].

In wireless sensor networks, Sensor nodes have limited resources, so we need to utilize them efficiently while transmitting the data. It is a challenge task to design energy-efficient WSN by considering the fault tolerance paradigm. The objective of our work is to aggregate sensed data in a secure manner using Quadratic-Minimum Spanning Tree (Q-MST) mechanism to enhance the fault tolerance situations. Q-MST topology is an extension of Minimum spanning tree (MST). Q-MST is an NP- Hard problem, moreover to solve Q-MST in the literature a meta-heuristic Artificial Bee Colony (ABC) algorithm is being proposed[5].

In wsn, sensor nodes are deployed randomly. So it may be possible that two or more sensor nodes are placed very close to each other, result in sensing the same data and may transmit to the same node. At receiving node data will be redundant so instead of transmitting the same data further it is better to aggregate the data before further transmission. Data aggregation is one of the technique to reduce redundant data for efficient energy utilization. Although it can be implemented by using any topology, but for this paper, we have used tree based topology. According to properties of tree, tree is neither self-loop nor circuit, in case any link fails graph become disconnected. To make it connected either we have to regenerate the whole graph or we need to find an alternative path such that graph become connected.

In many applications, secure transmission of data is most important. Security means reliable transmission of data from source to sink node. Providing secure data aggregation, while conserving energy is one of the most difficult tasks. To provide security, data needs to be encrypted and then transmitted. Aggregation of encrypted data is a challenging task. In this paper, we are going to present an algorithm to provide security requirements using Q-MST topology [1].

1.1 Motivation:

The objective of the work is to show the importance of Q-MST algorithm with conventional approach which is spanning tree method to find data aggregation. There are two approaches to find spanning tree which are prims and kruskal algorithm. The results also compared accordingly. In Sensor Networks energy is another crucial parameter which can be utilized more effectively in all the situations while data is transmitting from source to various sink nodes. In WSN data transmission should require security and elimination of redundancy while transmitting the data packets. This work focused on the two aspects in fault tolerant situations. The fault tolerant situation we can achieve by using Q-MST Meta heuristic. The results are also discussed accordingly in the results section. We have chosen energy is a crucial parameter. In this work we have shown the energy utilization in the fault tolerant situation Q-MST and in normal situation (using Prim's algorithm).

1.2 Quadratic-Minimum Spanning Tree (Q-MST) an Artificial Bee Colony (ABC) Algorithm:

Quadratic MST is an extension of Minimum Spanning Tree (MST) [5] problem in graphs to improve the effective utilization of resources. In sensor networks, the tree topology is one of the efficient topologies, from the energy utilization point of view. MST is one of the ways to implement tree topology. There are so many advantages of using MST in Sensor networks. The basic difference between MST and Q-MST is that in MST, each edge is associated with a cost. Where, in Q-MST cost is associated with each edge of the graph as well as with an ordered pair of distinct edges which were involved in the spanning tree. Objective of Q-MST is to find a minimum spanning tree that minimizes the total sum of the cost associated with individual edge and cost resulting from ordered pair of those edges which are present in the spanning tree. In tree topology, nodes are arranged in the form where sink node address as root and source nodes as leaves. If any link or any node fails, nodes below it get disconnected. To make connected, tree needs to be regenerated, this consumes lots of energy.

The advantage of using Q-MST over MST is that if link between two node breaks, then instead of regenerating the whole MST, first look up for an alternative path from those nodes between them link is broken. Cost of alternative path is calculated on the basis of ordered pairs of edges using formula:

$$\sum_{e_1 \neq e, e \in E} (c(e_1, e) + c(e, e_1)) + w(e) \tag{1}$$

In this work we have shown the simulation result which shows the improvements in the power consumption.

1.2.1 Artificial bee colony Algorithm :

Let the graph is a complete graph. An inter-cost matrix, is a cost matrix which contains the cost from one edge to every other edge. The diagonal element of the matrix represents the cost of individual edge. Figure 1 is a complete graph. Minimum spanning tree (MST) is formed in such a way that the cost associates with individual and cost of ordered pair of distinct edges should

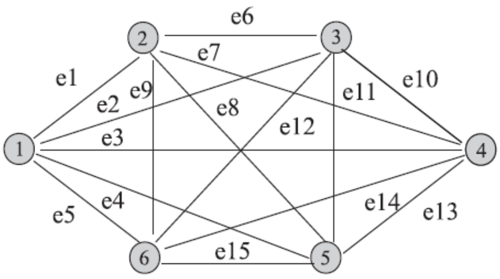


Fig. 1 Complete graph

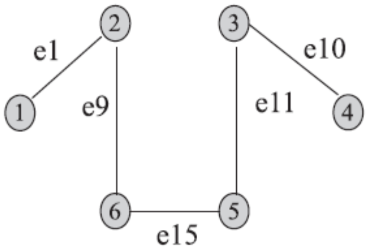


Fig. 2 Minimum Spanning Tree

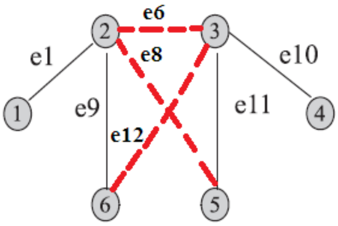


Fig. 3 Q-MST with possible alternative path

minimum. MST shown in figure 2. In case the link between node 5 and node 6 fails, the graph becomes disconnected. To make it connected either we need to regenerate the whole MST or to find the alternative path. Finding the alternative path is more energy efficient (As shown in figure 3): These are the possible alternative path as shown in in figure 3:
1. Connect node 3 and 6
2. Connect node 2 and 3
3. Connect node 2 and 5
Example:- Potential cost for node 2 to node 5 i.e edge e8 is calculated as:

C(e8) + C(e8,e11)+ C(e11, e8)+ C(e9,e8)+ C(e8, e9)+ C(e8, e1) +C(e1, e8)

1.2.2 Advantages of Q-MST:

Some of the merits of MST are: [4]
a. The Data Aggregation mechanism can be evaluated to reduce redundant data, to implement the data aggregation, we need the Minimal Spanning tree to implement the data aggregation.
b. The Q-MST mechanism would improve the network energy efficiency while establishing the path from the source node to sink node which leads to extended network life time.
c. Tree based topology is an efficient way to achieve effective flooding in sensor network.
d. Used in Converge cast to save energy. Converge cast means gathering of information towards a central node.
e. For Low Latency. Latency means the time required to send data from source node to sink node.

1.3 Secure data aggregation using Q-MST

Sensor networks are going more and more popular to provide efficient solutions for many situations. In sensor networks, an area is jointly monitored by thousands of nodes. In Sensor networks, sensor nodes senses the data, forward it to the base station via intermediate nodes. In large sensor network, a significant amount of data is generated; however the sensor nodes have limited resources like memory, computation power, battery energy, communication. Battery energy is the most important resource of sensor network [2].
To use the battery energy in an efficient way is the primary job of sensor network. Security is also an important parameter for transmitting sensitive data. Security along with data Aggregation is one of the techniques to efficiently use the energy of sensor nodes and transmitting data securely where overall result or a single result for a query is more important than information from individual node. The key idea of the aggregation is to integrate data coming from distinct sources en-route, which in return eliminate redundancy, minimizing the number of transmission and thus saving energy. Aggregation can be done by using any aggregator function: - MAX, MIN, COUNT, AVG, and SUM. Processing data for aggregation required less power as compare sending multiple packets having similar data. For security two-hop mechanism is used, ensuring data integrity and each wo hop and authentication at each hop. For secure data aggregation different mechanism are introduced in literature survey. In proposed work, Q-MST is implemented to enhance fault tolerance situation along with secure data transmission for transmitting sensitive data in an energy efficient way.

2 Literature Survey

Secure Data aggregation is crucial in WSN applications; the aggregation can save the energy as well as bandwidth in typical Wireless Environment. The Data Aggregation mechanism will reduce the number of collisions raised by different sensor nodes in the densely deployed sensor environment. The proposed work will ensure the fault tolerance in WSN with Q-MST approach which is Artificial Bee Colony (ABC) approach. The proposed work adopted two hop data integrity and one hop data verification with key pair generation mechanism [19][20].

2.1 Secure Data Aggregation:

Data aggregation [6, 7] becomes more challenging because of the deployment of sensor nodes in the uncongenial geographic area. So sensor nodes may get compromised by physical tampering. Fulfillment of Security requirements for data aggregation is a challenging task. Compromising node may result in false data. Thus, the processing and aggregation mechanisms should be resilient against attacks. For security both symmetric and asymmetric key encryption can be used. The main issue is that asymmetric key encryption provides more security, but more computation is also required. In symmetric key encryption less computation is required, but security is less comparatively.

2.1.1 Security Requirements:

If we talk about security that means the given security requirements need to be fulfilled [6, 7, 8].
Data Confidentiality: Data confidentiality ensures that data is known to authorize users only. If any unauthorized user comes to know the data, confidentiality loses. To ensure data confidentiality hop-by-hop and end-to end security methods are used. In hop-by-hop is known to all intermediate nodes which are authorized but in case if any intermediate node captured then data confidentiality loses. In end-to-end data is known to the source node and sink node only, data is encrypted at source node by using the shared key between sink node and source node. No intermediate node is allowed to access the data because secret key is known to source and sink node but forward to next node.
Data Integrity: It assures that the content of the data packet is not changed in between the transmission. Confidentiality is not enough to provide the complete security. It might be possible that data is lost because of behavior of the wireless environment.
Data Authentication: It ensures that the data send by the node is that only for which it is claim to be. Any node can participate or add data only if it has a legitimate authentication key. If authentication is not implemented, an attacker can impersonate (take id of some other node) any node and can get access to sensitive data.

Data Freshness: It assures that data is recent and is not replayed. Its assurance protects data from replay attack. Instead of altering or access the data, passive adversary may take the encrypted data and replay them later on to interrupt the aggregation.
Data Availability: If resources are available, then legitimate user should be able to access it.

Secure Data aggregation can be done by using any one of the following protocols [8]:
1. End-to-End security
2. Hop-by-Hop Security
Many algorithms exist on the basis of these protocols.
In End-to End protocol, decryption is done only at the sink node. No intermediate node can decrypt the message received by neighboring node, because data is encrypted using the shared key between a source node and sink node only. So no other node can decrypt it till it does not have shared key. For data aggregation End-to-End security is provided by using privacy homomorphism.
In hop-by-hop protocol, decryption is possible at any intermediate node. Many algorithms exist which are implemented to provide hop-by-hop security. For hop- by-hop security, pairwise key is generated between hops, which are consecutive and data need to be sent. So that using that pairwise key data can be decrypted. Data Integrity is one of the difficult tasks among all. In data aggregation of each node data is aggregated, so at each node it should be checked whether received data is same or altered. So checking this at all nodes for all received packets in an energy efficient way is very challenging. Many Protocols exist to provide these requirements with some benefits and some disadvantages .

2.2 Existing Algorithms:

S.Ozdenir,[9] Concealed Data Aggregation in Heterogeneous Sensor Networks using Privacy homomorphism paper deals with data aggregation on the encrypted data. Generally for data aggregation each parent node first decrypt the data sent by child node, apply some aggregation function and then decrypt it again and forward. At each node encryption and decryption is needed which is energy consuming method. So instead of this an alternative method is proposed which provide end-to-end confidentiality. Privacy homomorphism is one the methods to provide end-to-end data confidentiality and aggregation together. In Privacy homomorphism method encryption transformation is directly done on the encrypted data. Advantage of this method is that it provides end-to-end confidentiality. Disadvantage is that many sensor nodes suffer even if a single aggregator node is compromised. This method is suitable only for heterogeneous network only; because in homogeneous network some sensor nodes are taken as aggregator nodes (consume more energy), so need to select the aggregator node periodically which results small network lifetime.

A. Mahimkar, T. S. Rappaport,[10] SecureDAV: A Secure Data Aggregation and Verification Protocol for Sensor Networks Paper deals with the problem of aggregating, verifying the secure data and to make communication reliable. In this paper ECC (Elliptic curve cryptography) is being used for key management. To solve the ECC there is no exponential algorithm. So in ECC even smaller length key can provide same level of security as in RSA and DSA (with longer length key). Advantage of this algorithm is that in ECC smaller key size is being used, computation is faster. Each sensor node will generate a public-private key with cluster so confidentiality is maintained here. Disadvantage is that if cluster head is compromised then data of all sensor nodes of the cluster get compromised, as sensor nodes within a cluster communicate with its cluster head only. For data validation communication cost is high and only AVG aggregation function is supported.

L. Hu, D. Evans,[11] Secure aggregation for wireless networks Paper is to ensure that a faulty node can misdirect its own data. It should be capable of making any wrong claim about the reading of another node. Some assumptions are made for this approach: Sensor node can communicate with all sensor nodes within sensor field via one-hop. Second assumption is that each node is loaded with the shared symmetric key with base station. Third assumption is no two compromised nodes should be consecutive. Asymmetric algorithm for message authentication is not useful because of limited resources. So TESLA protocol is used here. In this approach a tree topology is used. Disadvantage of this approach is that wherever a invalid MAC is detected, node will omit two downward node from network. There is no mechanism to identifying the actual origin of packet. It is assumed that sink node can reach to all sensor nodes, using one-hop. This assumption is barely satisfied if network size is increases. This approach is suitable if network is deeper instead of many nodes at a single level, nodes are divided into levels. Number of nodes at a level is less. Key going to be used in MAC computation is periodically updated by sink node for each sensor node, which is an energy consuming task. If two consecutive nodes get compromised then it may not be possible to detect if any modification is attempted. As verification is done at sink node, final value is rejected after communicate up to sink node.

Y. Yang, X. Wang, S. Zhu, G. Cao,[14] SDAP: A secure hop-by-hop data aggregation protocol for sensor networks, paper is based on hop-by-hop secure aggregation have capability to tolerate more than one compromised node. This Paper is divided into two phase: In first phase all nodes are randomly grouped and form logical subtrees. For each group a leader node is selected. Periodically the leader node is changed. In second phase hop-by-hop data aggregation is performed for each group by its group leader. Using pair-wise shared key with parent node, leaf node encrypt its id, reading value, count and query. An extra bit flag is taken, to represent whether to aggregate result any more or not. If flag value is 1, intermediate node transmits to base station via its parent node without applying any aggregation function. If flag value is 0, intermediate node decrypt received message using pair-wise secret shared key. Disadvantage is that verification is done at sink node only, if false data detected final aggregated value is discarded after communicating up to sink node. It is not distributed so the verification process delayed until all data is received by sink node.

3 Proposed Work

Data aggregation is a crucial for increasing the network lifetime. The objective of this work is to provide data integrity in an energy efficient way for data aggregation.
There are two main problems in the existing algorithms of data aggregation: [12]
-Delayed authentication as verification done at the sink node only. Resulting in energy wastage in transmission of data packets which are not from authorized user.
-In case, any malicious or faulty node detected, sink node discards the whole aggregated data, which results loss of correct data also.

The proposed work mainly concentrating on two aspects which are fault tolerant and secure data Aggregation in WSN environment. We have used Q-MST to enhance the fault tolerance and SEDAN is a light weight protocol with the following steps:
1) Key establishment for two hop data aggregation.
2) Data authentication.
3) One-hop data integrity verification.
4) Authentication of aggregated data.
5) Two-hop data integrity verification.

3.1 Proposed Algorithm:

In the literature various kinds of security mechanisms are available to improve the performance of data aggregation. The proposed method is based on two hop verification technique [12]. This work is using the existing four kinds of key mechanisms.

Each node generates 4 different kinds of pairwise keys in Key establishment for two hop data aggregation phase:
a. One hop pairwise key of a node with its parent node.
b. One hop pairwise key of a node with each child node

c. Two hop pairwise key of a node with grandparent node.
d. Two hop pairwise key of a node with each grandchild node.
The following relation for aggregation function should verify at each node:

$$f(d_1, d_2, d_3, d_4, ...d_n) = f(d_1, (d_2, d_3, d_4), .d_n) \tag{2}$$

Terminology:
i = a node
P_i=Parent of node i
GP_i=Grandparent of node i
ID_i=Identity of node i
N_i=Nonce value of node i
d_i=Data packet of node i
K_{i,p_i}=Key shared between node i and its parent
K_{i,GP_i} =Key Shared between node i and its grandparent node

3.1.1 Pair-wise Key Generation Phase:

Each intermediate node calculates one hop MAC and two hop MAC. To perform this step each node requires a shared key with one hop and two hop node. For calculating one hop and two hop pairwise keys, a pairwise key generation algorithm is used [19]. In our paper for generating pairwise key we are going to use this concept. In this paper pairwise key generation mechanism is based on following phases:-
1. Initialization phase
2. Share-key discover

a. In first phase i.e. Initialization phase is done before the deployment of sensor nodes into the field. Key Distribution server generates a large pool of keys with their identifier, these keys are also known as seed key.

b. A random number l is generated by each sensor node which is known as the security level of sensor node.

c. Hash function is performed l (security level) times on m selected seed keys to generate m new keys, also known as derivative keys. After performing these steps neighbouring nodes share ID and security level.

d. Second Phase i.e. share key phase let the security level of the sensor node x is lx and for sensor node y is ly . If key Kx of sensor node x and the key Ky of sensor node y has a common key identifier, then their keys belongs to same family. To share secret key security level of sensor node is used and computed as follows:

Case 1: lx >ly . In this case, Kx=H $^{(\text{lx-ly})}$(Ky) And So sensor node x can use key Kx as their pairwise key Kxy The sensor node y compute the pairwise key Kxy =H $^{\text{(lx-ly)}}$(Ky).

Case 2: lx < ly. In this case, Ky=H $^{\text{(ly-lx)}}$(Kx) And So sensor node y can use key Ky as their pairwise key Kxy. The sensor node x compute the pairwise key Kxy=H $^{\text{(ly-lx)}}$(Kx)

Case 3: lx =ly . In this case, it is obvious Kx =Ky So sensor node x can use Kx as the pairwise key Kxy and sensor node y use Ky as the pairwise key Kxy.

Using these keys each node has the capability to check integrity at two-hop neighbor. In case, an invalid or corrupted data is received, it is discarded immediately, with no further need to forward it to the sink. Useless transmission of false data is avoided; therefore sensors energy resources are saved. After key generation following steps are performed for verification of authentication at each hop and data integrity at two-hop.

3.2 Data Authentication:

When a node i want to send its data di , It sends to its parent the following packet:

$$\begin{aligned} &ID_i, N_i, d_i, MAC(K_{i,P(i)}, N_i||d_i) \\ &ID_i, N_i, d_i, MAC(K_{i,GP(i)}, d_i||N_i) \end{aligned} \tag{3}$$

3.2.1 One-hop data Integrity Verification:

On every one hop transaction a node j from a child, it verifies the OHM to validate the origin of the packet, and stores the rest of the information: ID_{child}, N_{child}, d_{child}, THM_{child}.

3.2.2 Authentication of aggregated data:

Upon receiving the data of all its children, the node j computes the aggregation value over the childrens data. Since f(d_{child1}, d_{child2}, d_{child3},..., d_{childn}) will represent the data of j , the node must calculate it OHM and THM using the equations 3.

Steps:

1. Key establishment phase: Above given four different kinds of keys are generated for each node in the network.
2. If a node has some data to transmit, it will encrypt using one-hop pairwise shared key so that only a parent node is able to read the message. Using two-hop pair wise shared key also the data is encrypted so that grandparent node can verify the message. Let i be a node, following packets are forwarded by

each leaf node. Node i calculate MAC using one-hop pairwise share key i.e. one hop key (OHM) and two- hop pair wise shared key (THM).

$$\begin{aligned} ID_i, N_i, d_i, MAC(K_{i,P(i)}, N_i \| d_i) \\ ID_i, N_i, d_i, MAC(K_{i,GP(i)}, d_i \| N_i) \end{aligned} \tag{4}$$

3. After receiving the packet from the child node, the parent node will authorize the sender. It store the information as in:- ID_{child}, N_{child}, d_{child}, THM_{child}. Receiver will generate the MAC for received data and compare it with received MAC for verification. Then aggregate its own data and received data, generate MAC for aggregated data using one-hop and two-hop pairwise key. Along with aggregated MAC it also send MAC for its own sensed data. Each intermediate node will forward the following packets: Let j is parent of node i.

$$\begin{aligned} ID_i, N_i, d_i, MAC(K_{i,GP(i)}, d_i \| N_i) \\ ID_j, N_j, d_j, MAC(K_{j,P(j)}, N_j \| d_j) \\ MAC(K_{j,P(j)}, f(d_i, d_j) \| N_j) \\ MAC(K_{j,GP(j)}, N_j \| f(d_i, d_j) \end{aligned} \tag{5}$$

4. The Grandparent node (i.e parent of node j) will verify the received THM of its grandchild node, OHM of aggregated data sent by its child node, OHM of its childs individual data. After individual nodes MAC verification, aggregation on received data is performend and verified with received MAC. If both, the MACs are same, means the message is not altered between the transmission and forward further. If both are not same that means data is altered and is discarded immediately.

5. Each intermediate node will aggregate the data and calculate both OHM and THM . Thus, integrity ensured at second hop that no node is reporting any false data about another node. This process will be continued until the data reached at base station.

6. If any link between the two node break or any node fails, then instead of regenerating the whole MST, an alternative path is searched and on the basis of cost any one of the path is selected and keys are shared between them. The same procedure will be continued [5].

Link Failure between C and D Sensor nodes: As shown in Figure 5 if a link fails, the graph becomes disconnected. To make it connected either need to regenerate the MST or to find the alternative path using Q-MST. Quadratic-Minimum Spanning Tree (Q-MST) mechanism to enhance the fault

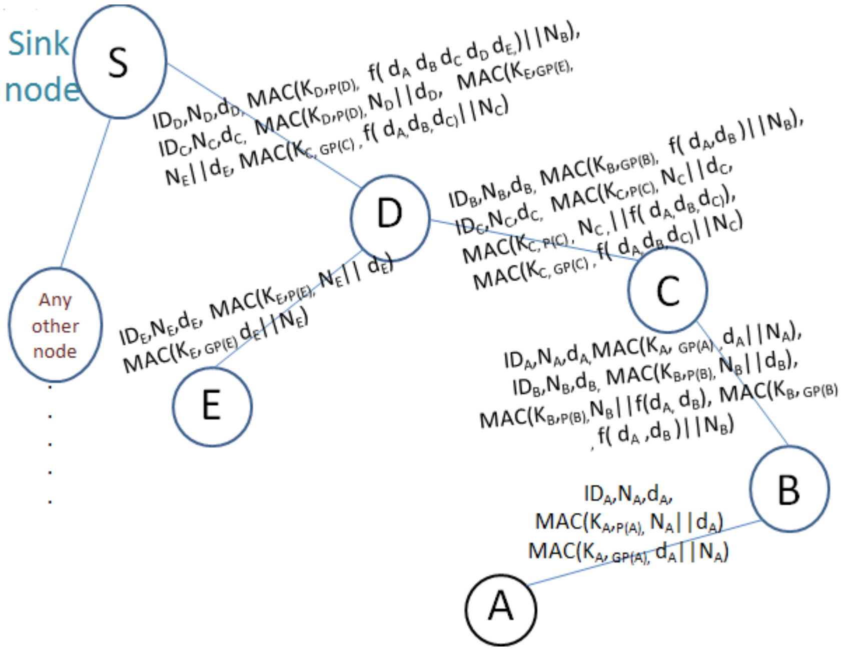

Sink node
S
$ID_D, N_D, d_D, MAC(K_{D,P(D)}, f(d_A\ d_B\ d_C\ d_D\ d_E)||N_B),$
$ID_C, N_C, d_C, MAC(K_{D,P(D)}, N_D||d_D, MAC(K_{E,GP(E)},$
$N_E||d_E, MAC(K_{C,GP(C)}, f(d_A, d_B, d_C)||N_C)$
D
$ID_B, N_B, d_B, MAC(K_{B,GP(B)}, f(d_A, d_B)||N_B),$
$ID_C, N_C, d_C, MAC(K_{C,P(C)}, N_C||d_C,$
$MAC(K_{C,P(C)}, N_C||f(d_A, d_B, d_C)),$
$MAC(K_{C,GP(C)}, f(d_A, d_B, d_C)||N_C)$
C
Any other node
$ID_E, N_E, d_E, MAC(K_{E,P(E)}, N_E||d_E)$
$MAC(K_{E,GP(E)}\ d_E||N_E)$
E
$ID_A, N_A, d_A, MAC(K_{A,GP(A)}, d_A||N_A),$
$ID_B, N_B, d_B, MAC(K_{B,P(B)}, N_B||d_B),$
$MAC(K_{B,P(B)}, N_B||f(d_A, d_B), MAC(K_{B,GP(B)}$
$, f(d_A, d_B)||N_B)$
B
$ID_A, N_A, d_A,$
$MAC(K_{A,P(A)}, N_A||d_A)$
$MAC(K_{A,GP(A)}, d_A||N_A)$
A

Fig. 4 Keys need to be exchanged

tolerance situations in an energy efficient way by aggregating data. Q-MST topology is an extension of Minimum spanning tree (MST). Q-MST is an NP-Hard problem; In the literature to solve Q-MST in the literature a heuristic Artificial Bee Colony (ABC) algorithm is being proposed. In this approach all the potential costs are evaluated for alternate edge in three phases which are scout, onlooker and employee bee phase. The three phases could find the alternate edge in place of failure link, The Q-MST is an implementation of three phase algorithm which could give an alternate edge in place of failure link. The less resource edge could be identified and replaced with the high resourceful edge. Let the link between node C and D fails. Possible paths to make connected graph Q-MST (Artificial bee colony), are B to D, B to E, and B to S, C to E and C to S as shown Figure 5. In Q-MST, alternative path is searched between the adjacent nodes of the link failure node.

3.3 Security Analysis:

Data Authentication: One-Hop verification is enough to authenticate a node. Every node is authenticated by its parent node. One secret key is shared between a node and its parent node. When a node wishes to transmit data packet, it first decrypts the message and then transmits. Before accepting the data, parent node first affirms validity of the node.

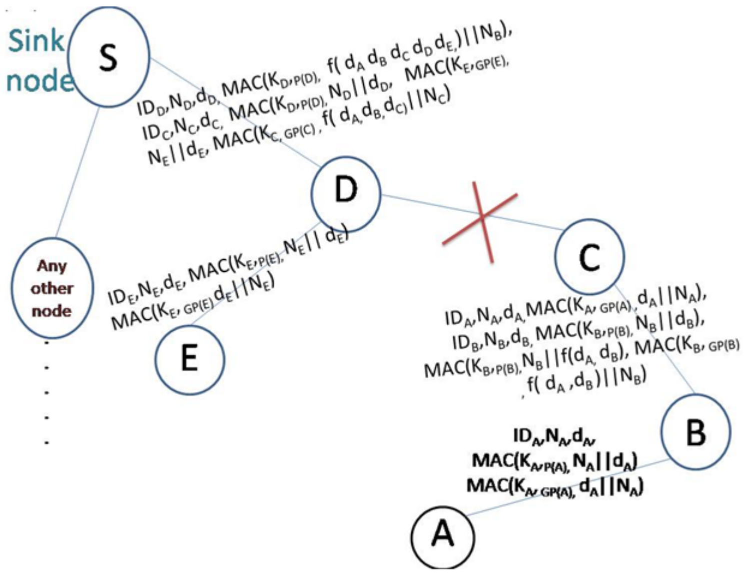

Sink node
S
$ID_D, N_D, d_D, MAC(K_{D,P(D)}, f(d_A\, d_B\, d_C\, d_D\, d_E)||N_B)$,
$ID_C, N_C, d_C, MAC(K_{D,P(D)}, N_D||d_D, MAC(K_{E,GP(E)}, N_E||d_E, MAC(K_{C,GP(C)}, f(d_A, d_B, d_C)||N_C)$
D
Any other node
$ID_E, N_E, d_E, MAC(K_{E,P(E)}, N_E||d_E)$
$MAC(K_{E,GP(E)}\, d_E||N_E)$
E
C
$ID_A, N_A, d_A, MAC(K_{A,GP(A)}, d_A||N_A)$,
$ID_B, N_B, d_B, MAC(K_{B,P(B)}, N_B||d_B)$,
$MAC(K_{B,P(B)}, N_B||f(d_A, d_B), MAC(K_{B,GP(B)}, f(d_A, d_B)||N_B)$
B
ID_A, N_A, d_A,
$MAC(K_{A,P(A)}, N_A||d_A)$
$MAC(K_{A,GP(A)}, d_A||N_A)$
A

Fig. 5 Link Failure

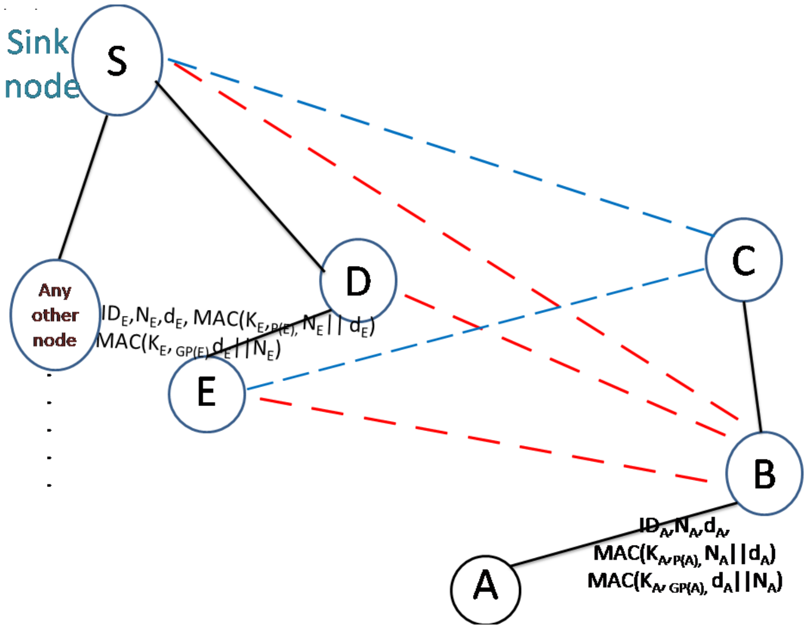

Sink node
S
Any other node
D
$ID_E, N_E, d_E, MAC(K_{E,P(E)}, N_E||d_E)$
$MAC(K_{E,GP(E)}\, d_E||N_E)$
E
C
B
ID_A, N_A, d_A,
$MAC(K_{A,P(A)}, N_A||d_A)$
$MAC(K_{A,GP(A)}, d_A||N_A)$
A

Fig. 6 Alternative paths using Q-MST approach

Data Integrity: Data integrity means to verify that data in which we have to find data is altered or not between the transmission. For the proposed work two-hop verification key mechanism is performed to ensure that the data is not altered in between the transmission.

Data Confidentiality: A node shares key with parent node and the grandparent node, but both keys are different. Parent node cannot decrypt the message which is encrypted by using the key shared between node and grandparent.
At each node, data are authenticated, and integrity is ensured by two-hop verification method. Data is known to authorize nodes (who know their pairwise key) only, so confidentiality also lies. The proposed work also concentrates on fault tolerant situation the following are some of the advantages:

Energy Efficient: In Q-MST, there are no cycles and adjacent nodes are also less compare to graph. All nodes are sending and receiving approximately equal number of packets. So energy is balanced, no delayed authentication and integrity. Also, energy is not wasted in transmitting the invalid data. In case a link is failed in between two nodes then no need to reconstruct the whole MST again. As Q-MST will find the alternative path would be searched, the path having minimum cost is selected as alternative.

Scalable: Each node has to store the pairwise key with their two hop nodes. All nodes are arranged in the form of a tree. Even if nodes are increased, all nodes instead of belonging to a single node will belong to different parent nodes. The energy of the entire network is still balanced. Thus it promotes scalability.

Impersonation attack: It is detected and data from malicious node is discarded.

Network lifetime: Overall energy consumption decreases hence increase in network lifetime.

Collision: Each node is receiving data from its adjacent nodes which are lesser in number. So the probability that nodes are transmitting data to a node at the same time is less. Hence the probability of collisions decreases.

Storage: In Q-MST numbers of nodes connected to single node are less than as in normal tree. So a node requires to store one-hop shared secret key with its parent and child and two-hop shared secret key grandparent and grandchildren. Needs small amount of data is to be stored.

Sensornode.jpg

Fig. 7 Generated WSN

4 Simulation results of proposed work

In the proposed work we have used MATLAB software to implement the proposed work based on the previous work done by the authors in the same field. There are variety of simulators is also developed by MATLAB, it is an application oriented simulator for WSN. The MATLAB can support modular approach which can sense of environment which is easy when compared to other programming language. The main purpose of choosing MATLAB is to know the performance of Q-MST approach in fault tolerant scenario. Therefore in this work we have used MATLAB in place of NS2, Qualnet or any other kind of WSN Simulators. MATLAB developed by MathWork. The functions provide solutions to mathematical problems including matrix algebra, complex arithmetic, linear systems, differential equations, and other kind of equations. The simulation results we can put easily in MATLAB Workspace [20].
We have created WSN with the given number of sensors and sink nodes [17, 18].
Number of required sensor nodes are entered through user interface. As shown in Figure 7. On the basis of distance nearest node from sink node is determined and treated as the root of Q-MST. If Link fails between node 3 and node 4 then instead of regerating MST alternative path is find out to make graph connected using Q-MST. As shown in Figure 8, Possible paths are:
a) Path between node 2 and 15
b) Path between node 15 and 3
c) Path between node 2 and 4
In case of link failure, either we need to regenerate the MST or to use Q-MST to find the alternative path to make graph connected. Figure 9 shows the time required in regenerating the MST or to find alternative path using Q-MST. Every node needs to store information about its neighbours. Regeneration of MST requires to find neighbouring node for each node of the MST, but in Q-MST only a few nodes need to be updated. Figure. 10 shows the time consumption in finding neighbouring node in case of regeneration of MST and updating neighboring nodes of new link in Q-MST Nodes between which link are changed need to exchange a shared secret key and to remove previous secret keys. For Q-MST no need to share a secret key for whole network key is shared for those only between which new path is created. Below Figure 11 shows the time comparison between sharing secret key in case of Q-MST and in case of regeneration of the MST.

Energy is an important resource of sensor networks. Initially all nodes are

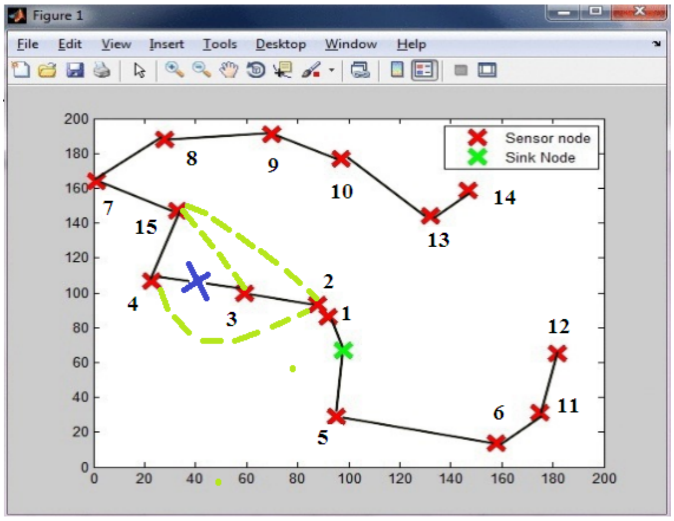


Fig. 8 Link failure between node 2 and 4 with possible alternative path

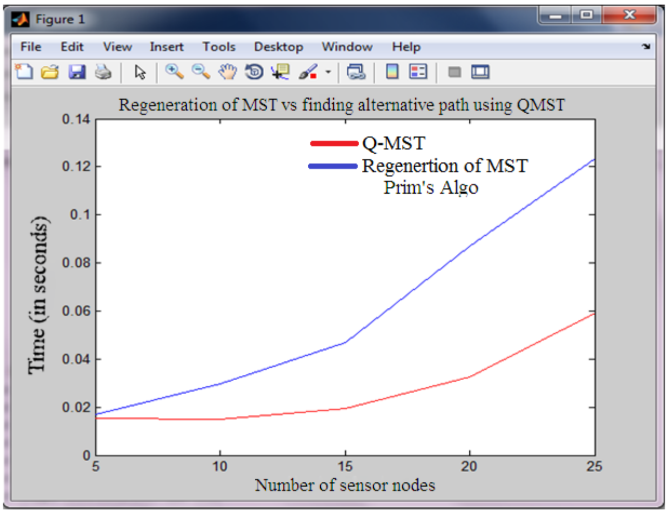


Fig. 9 Time required to regenerate the MST vs finding alternative path using ABC

given the same amount of energy. If any link failure is there, we need to regenerate the whole MST or to find the alternative path in both cases energy would be comsumed. Figure 12 shows Q-MST approach will consume less energy while compared to standard MST.
Let an existing path is replaced with alternate one, as per the security point of view key exchanges are required with new node which is add to in place of weaker node. But in case of regeneration using MST whole network nodes

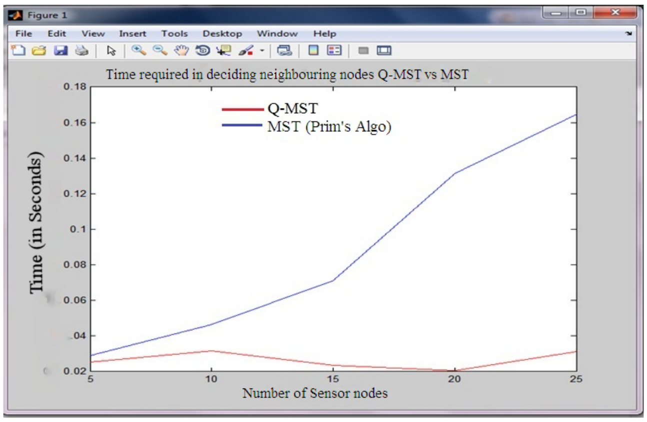


Fig. 10 Time required in finding neighboring nodes in Q-MST vs MST

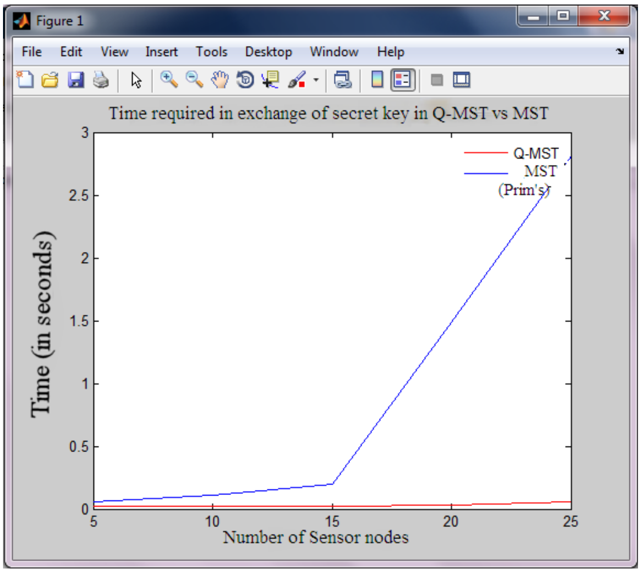


Fig. 11 Time required in key exchange using MST and Q- MST

need to be share key again so more energy consumed, Shown in Figure 13.

If in case more than one link failure happens simultaneously, Artificial bee colony algorithm still works and alternative paths for both link failure are determined. As Shown in Figure 14.
All possible alternative paths for two link failure are shown in figure 15.

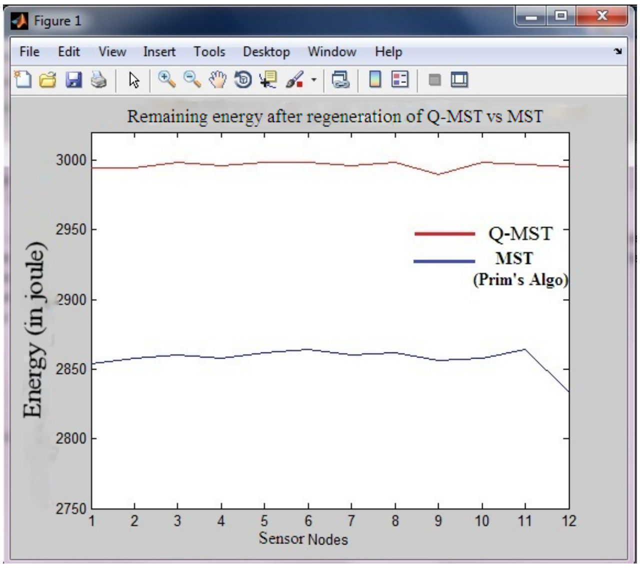


Fig. 12 Remaining energy after consumption in case of MST and Q-MST

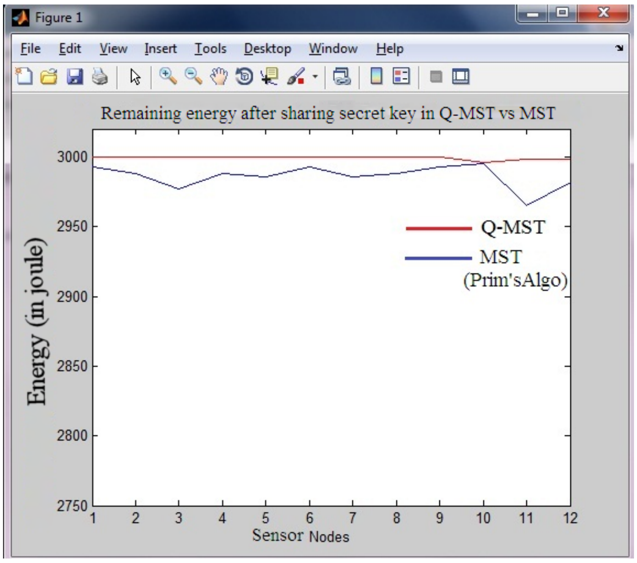
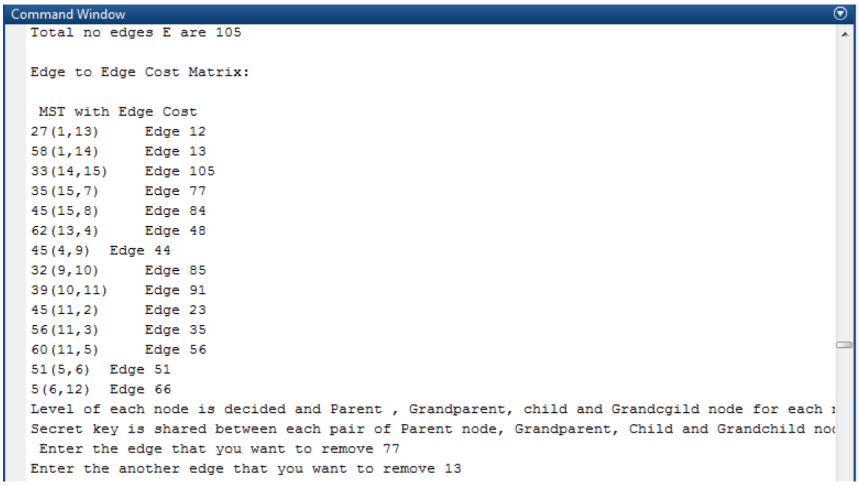


Fig. 13 Remaining Energy after consumption for Sharing Secret Key in Q-MST vs MST

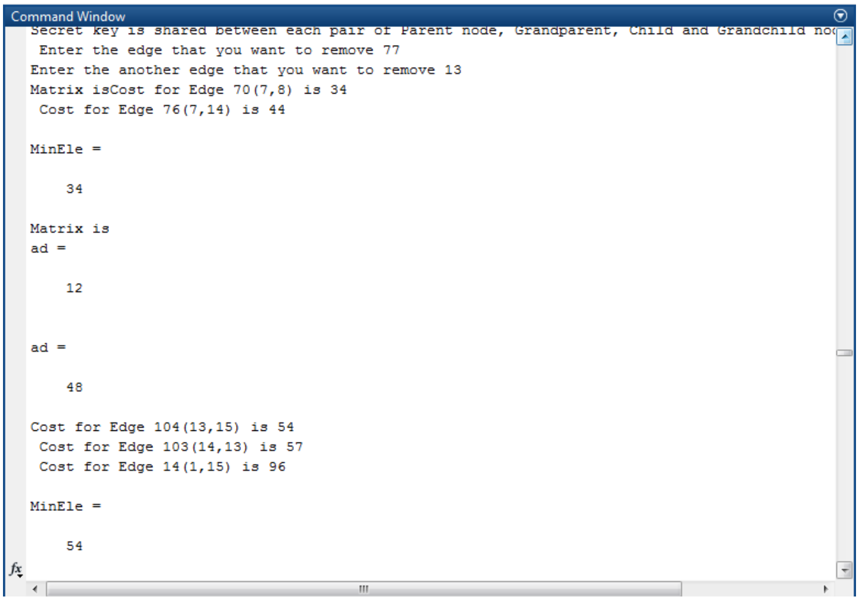
5 Conclusion:

Sensor nodes are having limited energy power. It is a challenge to use energy in an efficient way. Here, we conclude that data aggregation is one of the methods to utilize energy in an efficient way, reduces communication overhead, increase network lifetime etc. Security is another important aspect of WSN. In real



```
Total no edges E are 105

Edge to Edge Cost Matrix:

 MST with Edge Cost
27(1,13)      Edge 12
58(1,14)      Edge 13
33(14,15)     Edge 105
35(15,7)      Edge 77
45(15,8)      Edge 84
62(13,4)      Edge 48
45(4,9)  Edge 44
32(9,10)      Edge 85
39(10,11)     Edge 91
45(11,2)      Edge 23
56(11,3)      Edge 35
60(11,5)      Edge 56
51(5,6)  Edge 51
5(6,12)  Edge 66
Level of each node is decided and Parent , Grandparent, child and Grandcgild node for each
Secret key is shared between each pair of Parent node, Grandparent, Child and Grandchild no
 Enter the edge that you want to remove 77
Enter the another edge that you want to remove 13
```

Fig. 14 A Scenerio, more than one link failure



```
Secret key is shared between each pair of Parent node, Grandparent, Child and Grandchild no
 Enter the edge that you want to remove 77
Enter the another edge that you want to remove 13
Matrix isCost for Edge 70(7,8) is 34
 Cost for Edge 76(7,14) is 44

MinEle =

    34

Matrix is
ad =

    12


ad =

    48

Cost for Edge 104(13,15) is 54
 Cost for Edge 103(14,13) is 57
 Cost for Edge 14(1,15) is 96

MinEle =

    54
```

Fig. 15 All possible alternative paths in case of more than one link failure

applications, data aggregation without security is impractical. The Artificial bee colony algorithm is implemented in Q-MST to find the alternative paths in case of any link failure. Using Q-MST is more energy efficient, less time consuming compared to regenerating the whole MST. The Artificial bee colony algorithm also works correctly in case of more than one link failure. Security is another important aspect in WSN for which SEDAN algorithm is used for two hop data integrity.

References

1. I. F. Akyildiz, W. Su., Y. Sankarrasubramaniam, and E. Cayirci, A survey on sensor networks, IEEE Communication Magazine, 40, 104-112(2002).
2. Sushruta Mishra, and Hiren Thakka, Features of WSN and Data Aggregation techniques in WSN: A Survey, International Journal of Engineering and Innovative, 1, 2002.
3. Z. Eskandari, M. Hossien Yaghmaee, and A. Mohajerzadeh, Wireless sensor networks:A Survey, ELSEVIER computer network, 2002.
4. S. Upadhyayula, and S. K. S. Gupta, Spanning tree based algorithms for low latency and energy efficient data aggregation enhanced converge cast (DAC) in wireless sensor networks, ELSEVIER Ad Hoc Networks, 5(5), 626648(2007).
5. Shyam Sundar, and Alok Singh, A swarm intelligence approach to the quadratic minimum spanning tree, ELSEVIER Information science, 180(17), 31823191(2010).
6. Adrian Perring, John Stankovic, and David Wagner, Security in Wireless Sensor Networks, Communications of the ACM, 47(6), 2004.
7. S. Ozdemir, and Y. Xiao, Secure data aggregation in wireless sensor networks: A comprehensive overview, ELSEVIER Computer Networks, 52(12), 20222037(2009).
8. DJosna Jose, Joyce Jose, and Fijo Jose, A Survey on Secure Data Aggregation Protocols in Wireless Sensor Networks, Int. Journal Computer Applications, 55(18), 2012.
9. Suat Ozdenir, Concealed Data Aggregation in Heterogeneous Sensor Networks using Privacy homomorphism, IEEE Conf. on pervasive services, 163-168(2007).
10. A. Mahimkar, and T. S. Rappaport, SecureDAV: A Secure Data Aggregation and Verification Protocol for Sensor Networks, Proc. IEEE Global Telecomm. Conf. (Globecom), 4, 2175-2179(2004).
11. L.Hu, and D.Evans, Secure Aggregation for Wireless Networks, Proc. Symp.Applications and the Internet Workshops, 2003.
12. M. Bagaa, Y. Challal, A Ouadjaout , N. Lasla, and N. Badache, Efficient data aggregation with in-network integrity control for WSN, ELSEVIER Information J. Parallel Distrib. Comput, 72(10), 1157-1170(2012).
13. C. Lin, G. Wu, F. Xia, M. Li, L. Yao, and Z. Pei, Energy efficient ant colony algorithms for data aggregation in wireless sensor networks, ELSEVIER Information Journal of computer science and system, 78(6), 1682-1702(2012).
14. Y. Yang, X. Wang, S. Zhu, and G. Cao, SDAP: A secure hop-by-hop data aggregation protocol for sensor networks, Proceedings of The ACM International Symposium on Mobile Ad Hoc Networking and Computing, 356-367(2006).
15. Yingpeng Sang, Hong SHen, Y. Inoguchi, Yasuo Tan, and Naixue Xiong, Secure Data Aggregation in Wireless Sensor Networks: A survey, IEEE Proceedings Parallel and Distributed Computing, Applications and Technologies, 315-320(2006).
16. W. Du, J. Deng, Y.S. Han, and P. K. Varshney, A witness based approach for data fusion assurance in wireless sensor networks, IEEE Global communication Conference (GLOBECOM), 3, 1435-1439(2003).
17. User Guide Mathworks, MATLAB and SIMULATION.
18. http://www.mathworks.in/help.html.
19. Mu Kun and Li Li , An Efficient Pairwise Key Predistribution Scheme for Wireless Sensor Networks , JOURNAL OF NETWORKS,9(2), 2014 .
20. Qutaiba Ali, Akram Adulmaowjod, H. M. Mohammed Simulation performance study of wireless sensor network (WSN) using MATLAB, Iraq J. Electrical and Electronic Engineering, 7(2), 2011.

Click here to download author's picture & biography bio-data.docx

Nagaraju Aitha received M.Sc (Pure Maths) from the Central University of Hyderabad in 1998; M.Tech in Computer Science and Technology from University of Mysore in 2001, and Ph.D from Osmania University, Hyderabad. His research area includes Wireless Sensor Networks, Software Defined Networks (SDN) and Cloud Computing. He has published 25 conference papers 12 peer reviewed journals. He worked as various positions from 2002-12 in two engineering colleges affiliated to JNTU, Hyderabad. At present, He is working as Assistant Professor in Computer Science at Central University of Rajasthan since 2012. The University appointed as NKN Coordinator to perform the objectives of NKN. He has established a Remote Center to conduct online workshops using blended mode which is T10kT Project initiated by MHRD-IITB. He is School Board member for two Schools in Central University of Rajasthan.

Priyanka Nehra received B.Tech (CSE) from University of Rajasthan and M.Tech (CSE) with specialization Information Security from Central University of Rjasthan, Currently pursuing Ph.D under the Visvesvaraya Ph.D Scheme for Electronics and IT which is fully funded by **DeitY, Govt of India**, and Her research area includes Wireless Sensor Networks and Workflow Scheduling in Cloud Computing.

attachment to manuscript
Click here to download attachment to manuscript template1.tex
Click here to view linked References

```
This is pdfTeX, Version 3.14159265-2.6-1.40.16 (TeX Live 2015/W32TeX)
(preloaded format=pdflatex 2016.4.6)  22 JUN 2016 09:55
entering extended mode
 restricted \write18 enabled.
 %&-line parsing enabled.
**./template1.tex
(./template1.tex
LaTeX2e <2016/03/31>
Babel <3.9q> and hyphenation patterns for 81 language(s) loaded.

LaTeX Warning: File `example.eps' already exists on the system.
               Not generating it from this source.

(c:/TeXLive/2015/texmf-dist/tex/latex/base/fix-cm.sty
Package: fix-cm 2015/01/14 v1.1t fixes to LaTeX
(c:/TeXLive/2015/texmf-dist/tex/latex/base/ts1enc.def
File: ts1enc.def 2001/06/05 v3.0e (jk/car/fm) Standard LaTeX file
)) (c:/TeXLive/texmf-local/tex/latex/aries/springer/svjour3.cls
Document Class: svjour3 2010/11/25 v3.3
LaTeX document class for Springer journals
Class Springer-SVJour3 Info: extra/valid Springer sub-package (-> *.clo)
(Springer-SVJour3)           not found in option list of \documentclass
(Springer-SVJour3)           - autoactivating "global" style.
(c:/TeXLive/texmf-local/tex/latex/aries/springer/svglov3.clo
File: svglov3.clo 2009/12/18 v3.2 style option for standardised journals
SVJour Class option: svglov3.clo for standardised journals
)
LaTeX Font Info:    Redeclaring math symbol \Gamma on input line 147.
LaTeX Font Info:    Redeclaring math symbol \Delta on input line 148.
LaTeX Font Info:    Redeclaring math symbol \Theta on input line 149.
LaTeX Font Info:    Redeclaring math symbol \Lambda on input line 150.
LaTeX Font Info:    Redeclaring math symbol \Xi on input line 151.
LaTeX Font Info:    Redeclaring math symbol \Pi on input line 152.
LaTeX Font Info:    Redeclaring math symbol \Sigma on input line 153.
LaTeX Font Info:    Redeclaring math symbol \Upsilon on input line 154.
LaTeX Font Info:    Redeclaring math symbol \Phi on input line 155.
LaTeX Font Info:    Redeclaring math symbol \Psi on input line 156.
LaTeX Font Info:    Redeclaring math symbol \Omega on input line 157.
\logodepth=\dimen102
\headerboxheight=\dimen103
\betweenumberspace=\dimen104
\aftertext=\dimen105
\headlineindent=\dimen106
\c@inst=\count79
\c@auth=\count80
\instindent=\dimen107
\authrun=\box26
\authorrunning=\toks14
\titrun=\box27
\titlerunning=\toks15
\combirun=\box28
\c@lastpage=\count81
\rubricwidth=\dimen108
\c@section=\count82
```

```
\c@subsection=\count83
\c@subsubsection=\count84
\c@paragraph=\count85
\c@subparagraph=\count86
\spthmsep=\dimen109
\c@theorem=\count87
\c@case=\count88
\c@conjecture=\count89
\c@corollary=\count90
\c@definition=\count91
\c@example=\count92
\c@exercise=\count93
\c@lemma=\count94
\c@note=\count95
\c@problem=\count96
\c@property=\count97
\c@proposition=\count98
\c@question=\count99
\c@solution=\count100
\c@remark=\count101
\c@figure=\count102
\c@table=\count103
\abovecaptionskip=\skip41
\belowcaptionskip=\skip42
\figcapgap=\dimen110
\tabcapgap=\dimen111
\figgap=\dimen112
\bibindent=\dimen113
\@tempcntc=\count104
) (c:/TeXLive/2015/texmf-dist/tex/latex/graphics/graphicx.sty
Package: graphicx 2014/10/28 v1.0g Enhanced LaTeX Graphics (DPC,SPQR)
(c:/TeXLive/2015/texmf-dist/tex/latex/graphics/keyval.sty
Package: keyval 2014/10/28 v1.15 key=value parser (DPC)
\KV@toks@=\toks16
) (c:/TeXLive/2015/texmf-dist/tex/latex/graphics/graphics.sty
Package: graphics 2016/01/03 v1.0q Standard LaTeX Graphics (DPC,SPQR)
(c:/TeXLive/2015/texmf-dist/tex/latex/graphics/trig.sty
Package: trig 2016/01/03 v1.10 sin cos tan (DPC)
) (c:/TeXLive/2015/texmf-dist/tex/latex/latexconfig/graphics.cfg
File: graphics.cfg 2010/04/23 v1.9 graphics configuration of TeX Live
)
Package graphics Info: Driver file: pdftex.def on input line 95.
(c:/TeXLive/2015/texmf-dist/tex/latex/pdftex-def/pdftex.def
File: pdftex.def 2011/05/27 v0.06d Graphics/color for pdfTeX
(c:/TeXLive/2015/texmf-dist/tex/generic/oberdiek/infwarerr.sty
Package: infwarerr 2010/04/08 v1.3 Providing info/warning/error messages
(HO)
) (c:/TeXLive/2015/texmf-dist/tex/generic/oberdiek/ltxcmds.sty
Package: ltxcmds 2011/11/09 v1.22 LaTeX kernel commands for general use
(HO)
)
\Gread@gobject=\count105
))
\Gin@req@height=\dimen114
```

```
\Gin@req@width=\dimen115
) (./template1.aux

LaTeX Warning: Label `Ref9' multiply defined.


LaTeX Warning: Label `RefJ' multiply defined.


LaTeX Warning: Label `RefJ' multiply defined.


LaTeX Warning: Label `RefJ' multiply defined.


LaTeX Warning: Label `RefB' multiply defined.


LaTeX Warning: Label `RefJ' multiply defined.


LaTeX Warning: Label `RefJ' multiply defined.

)
\openout1 = `template1.aux'.

LaTeX Font Info:    Checking defaults for OML/cmm/m/it on input line 62.
LaTeX Font Info:    ... okay on input line 62.
LaTeX Font Info:    Checking defaults for T1/cmr/m/n on input line 62.
LaTeX Font Info:    ... okay on input line 62.
LaTeX Font Info:    Checking defaults for OT1/cmr/m/n on input line 62.
LaTeX Font Info:    ... okay on input line 62.
LaTeX Font Info:    Checking defaults for OMS/cmsy/m/n on input line 62.
LaTeX Font Info:    ... okay on input line 62.
LaTeX Font Info:    Checking defaults for OMX/cmex/m/n on input line 62.
LaTeX Font Info:    ... okay on input line 62.
LaTeX Font Info:    Checking defaults for U/cmr/m/n on input line 62.
LaTeX Font Info:    ... okay on input line 62.
LaTeX Font Info:    Checking defaults for TS1/cmr/m/n on input line 62.
LaTeX Font Info:    ... okay on input line 62.
Unknown journal: specify \journalname{<name of your journal>} in preambel

(c:/TeXLive/2015/texmf-dist/tex/context/base/supp-pdf.mkii
[Loading MPS to PDF converter (version 2006.09.02).]
\scratchcounter=\count106
\scratchdimen=\dimen116
\scratchbox=\box29
\nofMPsegments=\count107
\nofMParguments=\count108
\everyMPshowfont=\toks17
\MPscratchCnt=\count109
\MPscratchDim=\dimen117
\MPnumerator=\count110
\makeMPintoPDFobject=\count111
```

```
\everyMPtoPDFconversion=\toks18
) (c:/TeXLive/2015/texmf-dist/tex/generic/oberdiek/pdftexcmds.sty
Package: pdftexcmds 2011/11/29 v0.20 Utility functions of pdfTeX for
LuaTeX (HO
)
(c:/TeXLive/2015/texmf-dist/tex/generic/oberdiek/ifluatex.sty
Package: ifluatex 2010/03/01 v1.3 Provides the ifluatex switch (HO)
Package ifluatex Info: LuaTeX not detected.
) (c:/TeXLive/2015/texmf-dist/tex/generic/oberdiek/ifpdf.sty
Package: ifpdf 2011/01/30 v2.3 Provides the ifpdf switch (HO)
Package ifpdf Info: pdfTeX in PDF mode is detected.
)
Package pdftexcmds Info: LuaTeX not detected.
Package pdftexcmds Info: \pdf@primitive is available.
Package pdftexcmds Info: \pdf@ifprimitive is available.
Package pdftexcmds Info: \pdfdraftmode found.
) (c:/TeXLive/2015/texmf-dist/tex/latex/oberdiek/epstopdf-base.sty
Package: epstopdf-base 2010/02/09 v2.5 Base part for package epstopdf
(c:/TeXLive/2015/texmf-dist/tex/latex/oberdiek/grfext.sty
Package: grfext 2010/08/19 v1.1 Manage graphics extensions (HO)
(c:/TeXLive/2015/texmf-dist/tex/generic/oberdiek/kvdefinekeys.sty
Package: kvdefinekeys 2011/04/07 v1.3 Define keys (HO)
)) (c:/TeXLive/2015/texmf-dist/tex/latex/oberdiek/kvoptions.sty
Package: kvoptions 2011/06/30 v3.11 Key value format for package options
(HO)
(c:/TeXLive/2015/texmf-dist/tex/generic/oberdiek/kvsetkeys.sty
Package: kvsetkeys 2012/04/25 v1.16 Key value parser (HO)
(c:/TeXLive/2015/texmf-dist/tex/generic/oberdiek/etexcmds.sty
Package: etexcmds 2011/02/16 v1.5 Avoid name clashes with e-TeX commands
(HO)
Package etexcmds Info: Could not find \expanded.
(etexcmds)             That can mean that you are not using pdfTeX 1.50
or
(etexcmds)             that some package has redefined \expanded.
(etexcmds)             In the latter case, load this package earlier.
)))
Package grfext Info: Graphics extension search list:
(grfext)
[.png,.pdf,.jpg,.mps,.jpeg,.jbig2,.jb2,.PNG,.PDF,.JPG,.JPE
G,.JBIG2,.JB2,.eps]
(grfext)             \AppendGraphicsExtensions on input line 452.
(c:/TeXLive/2015/texmf-dist/tex/latex/latexconfig/epstopdf-sys.cfg
File: epstopdf-sys.cfg 2010/07/13 v1.3 Configuration of (r)epstopdf for
TeX Liv
e
))
SVJour3 warning: Missing \journalname
Missing character: There is no â in font cmbx5!
Missing character: There is no € in font cmbx5!
Missing character: There is no ¢ in font cmbx5!
LaTeX Font Info:    External font `cmex10' loaded for size
(Font)              <7> on input line 94.
LaTeX Font Info:    External font `cmex10' loaded for size
(Font)              <5> on input line 94.
```

```
LaTeX Font Info:    External font `cmex10' loaded for size
(Font)              <8> on input line 94.
LaTeX Font Info:    External font `cmex10' loaded for size
(Font)              <6> on input line 94.
Missing character: There is no â in font cmr5!
Missing character: There is no € in font cmr5!
Missing character: There is no ¢ in font cmr5!

SVJour3 Warning: Title too long for running head.
Please supply a shorter form with \titlerunning prior to \maketitle
[1{c:/TeXLive/2015/texmf-var/fonts/map/pdftex/updmap/pdftex.map}] [2]
! Undefined control sequence.
l.117 \sum_{\substack
                     {
The control sequence at the end of the top line
of your error message was never \def'ed. If you have
misspelled it (e.g., `\hobx'), type `I' and the correct
spelling (e.g., `I\hbox'). Otherwise just continue,
and I'll forget about whatever was undefined.

<complete.png, id=15, 364.36125pt x 197.98969pt>
File: complete.png Graphic file (type png)
<use complete.png>
Package pdftex.def Info: complete.png used on input line 126.
(pdftex.def)             Requested size: 200.39929pt x 108.89465pt.
<mstttt.png, id=16, 254.45062pt x 184.43906pt>
File: mstttt.png Graphic file (type png)
<use mstttt.png>
Package pdftex.def Info: mstttt.png used on input line 132.
(pdftex.def)             Requested size: 165.39093pt x 119.88396pt.
<QMST.png, id=17, 254.45062pt x 184.43906pt>
File: QMST.png Graphic file (type png)
<use QMST.png>
Package pdftex.def Info: QMST.png used on input line 138.
(pdftex.def)             Requested size: 152.67154pt x 110.66428pt.

Underfull \hbox (badness 10000) in paragraph at lines 124--148

 []

[3]
Underfull \vbox (badness 1845) has occurred while \output is active []

[4 <./complete.png> <./mstttt.png> <./QMST.png>] [5]
Underfull \hbox (badness 10000) in paragraph at lines 166--172

 []

[6]
Underfull \hbox (badness 10000) in paragraph at lines 173--179

 []
```

```
Underfull \hbox (badness 10000) in paragraph at lines 180--181

 []

[7]
Missing character: There is no â in font cmr10!
Missing character: There is no € in font cmr10!
Missing character: There is no □ in font cmr10!

Underfull \hbox (badness 10000) in paragraph at lines 182--184

 []


Underfull \hbox (badness 10000) in paragraph at lines 185--186

 []


Underfull \hbox (badness 10000) in paragraph at lines 188--189

 []

[8]
Underfull \hbox (badness 10000) in paragraph at lines 191--196

 []


Underfull \hbox (badness 10000) in paragraph at lines 197--204

 []


Underfull \hbox (badness 10000) in paragraph at lines 205--206

 []

[9]
Missing character: There is no â in font cmr10!
Missing character: There is no € in font cmr10!
Missing character: There is no ¦ in font cmr10!

Underfull \hbox (badness 10000) in paragraph at lines 216--226

 []


Underfull \hbox (badness 10000) in paragraph at lines 228--231

 []


Underfull \hbox (badness 10000) in paragraph at lines 232--233
```

```
 []


Underfull \hbox (badness 10000) in paragraph at lines 234--235

 []


Underfull \hbox (badness 10000) in paragraph at lines 236--237

 []


Underfull \hbox (badness 10000) in paragraph at lines 238--239

 []


Underfull \hbox (badness 10000) in paragraph at lines 240--241

 []

[10]
Underfull \hbox (badness 10000) in paragraph at lines 242--243

 []


Underfull \hbox (badness 10000) in paragraph at lines 244--245

 []


Underfull \hbox (badness 10000) in paragraph at lines 246--247

 []

Missing character: There is no â in font cmr10!
Missing character: There is no € in font cmr10!
Missing character: There is no ™ in font cmr10!

Underfull \hbox (badness 10000) in paragraph at lines 262--266

 []


Underfull \hbox (badness 10000) in paragraph at lines 262--266

 []

[11]
Underfull \hbox (badness 10000) in paragraph at lines 272--274
```

```
 []


Underfull \hbox (badness 10000) in paragraph at lines 286--288

 []


Underfull \hbox (badness 10000) in paragraph at lines 289--290

 []

<Algo1.png, id=58, 456.95718pt x 344.78813pt>
File: Algo1.png Graphic file (type png)
<use Algo1.png>
Package pdftex.def Info: Algo1.png used on input line 293.
(pdftex.def)             Requested size: 319.86784pt x 241.35004pt.
<linkfail.png, id=59, 478.78876pt x 392.96812pt>
File: linkfail.png Graphic file (type png)
<use linkfail.png>
Package pdftex.def Info: linkfail.png used on input line 300.
(pdftex.def)             Requested size: 287.27547pt x 235.78268pt.
<alternate.png, id=60, 722.7pt x 542.025pt>
File: alternate.png Graphic file (type png)
<use alternate.png>
Package pdftex.def Info: alternate.png used on input line 305.
(pdftex.def)             Requested size: 310.75494pt x 233.0662pt.

Underfull \hbox (badness 10000) in paragraph at lines 291--309

 []

[12]
Underfull \hbox (badness 10000) in paragraph at lines 310--314

 []


Underfull \hbox (badness 10000) in paragraph at lines 310--314

 []


Underfull \hbox (badness 10000) in paragraph at lines 310--314

 []

[13 <./Algo1.png>] [14 <./linkfail.png> <./alternate.png>]
Underfull \hbox (badness 10000) in paragraph at lines 315--316

 []


Underfull \hbox (badness 10000) in paragraph at lines 317--318
```

```
 []


Underfull \hbox (badness 10000) in paragraph at lines 319--320

 []


Underfull \hbox (badness 10000) in paragraph at lines 321--322

 []


Underfull \hbox (badness 10000) in paragraph at lines 323--324

 []


Underfull \hbox (badness 10000) in paragraph at lines 325--326

 []

<Sensornode.jpg, id=72, 427.5975pt x 382.42876pt>
File: Sensornode.jpg Graphic file (type jpg)
<use Sensornode.jpg>
Package pdftex.def Info: Sensornode.jpg used on input line 332.
(pdftex.def)             Requested size: 248.00671pt x 221.80882pt.
<sn.png, id=73, 426.09187pt x 325.96782pt>
File: sn.png Graphic file (type png)
<use sn.png>
Package pdftex.def Info: sn.png used on input line 340.
(pdftex.def)             Requested size: 251.39201pt x 192.31934pt.
<11.png, id=74, 463.7325pt x 354.57469pt>
File: 11.png Graphic file (type png)
<use 11.png>
Package pdftex.def Info: 11.png used on input line 347.
(pdftex.def)             Requested size: 250.4118pt x 191.46747pt.
<22.png, id=75, 561.59813pt x 362.10281pt>
File: 22.png Graphic file (type png)
<use 22.png>
Package pdftex.def Info: 22.png used on input line 353.
(pdftex.def)             Requested size: 241.48248pt x 155.70116pt.
<33.png, id=76, 434.37282pt x 385.44pt>
File: 33.png Graphic file (type png)
<use 33.png>
Package pdftex.def Info: 33.png used on input line 359.
(pdftex.def)             Requested size: 238.90578pt x 211.99265pt.
<44.png, id=77, 433.62pt x 382.42876pt>
File: 44.png Graphic file (type png)
<use 44.png>
Package pdftex.def Info: 44.png used on input line 365.
(pdftex.def)             Requested size: 238.49173pt x 210.33646pt.
<55.png, id=78, 431.36156pt x 378.66469pt>
```

```
File: 55.png Graphic file (type png)
<use 55.png>
Package pdftex.def Info: 55.png used on input line 370.
(pdftex.def)             Requested size: 237.24959pt x 208.26622pt.

Underfull \hbox (badness 10000) in paragraph at lines 328--375

 []


Underfull \hbox (badness 10000) in paragraph at lines 328--375

 []


Underfull \vbox (badness 10000) has occurred while \output is active []

[15]
Underfull \vbox (badness 3049) has occurred while \output is active []

[16 <./Sensornode.jpg>]
Underfull \vbox (badness 2126) has occurred while \output is active []

[17 <./sn.png> <./11.png>] [18 <./22.png> <./33.png>] <66.png, id=92,
584.93532
pt x 325.96782pt>
File: 66.png Graphic file (type png)
<use 66.png>
Package pdftex.def Info: 66.png used on input line 378.
(pdftex.def)             Requested size: 321.71541pt x 179.28284pt.
<77.png, id=93, 583.42969pt x 405.01312pt>
File: 77.png Graphic file (type png)
<use 77.png>
Package pdftex.def Info: 77.png used on input line 384.
(pdftex.def)             Requested size: 320.88731pt x 222.75789pt.
[19 <./44.png> <./55.png>]
Underfull \vbox (badness 10000) has occurred while \output is active []

[20 <./66.png> <./77.png>]
Missing character: There is no â in font cmr8!
Missing character: There is no € in font cmr8!
Missing character: There is no ‟ in font cmr8!
Missing character: There is no â in font cmr8!
Missing character: There is no € in font cmr8!
Missing character: There is no ‟ in font cmr8!
Missing character: There is no â in font cmr8!
Missing character: There is no € in font cmr8!
Missing character: There is no ‟ in font cmr8!
[21]
! Misplaced alignment tab character &.
l.497 Simulation    &
                       performance  study  of    wireless   sensor
network    ...
I can't figure out why you would want to use a tab mark
```

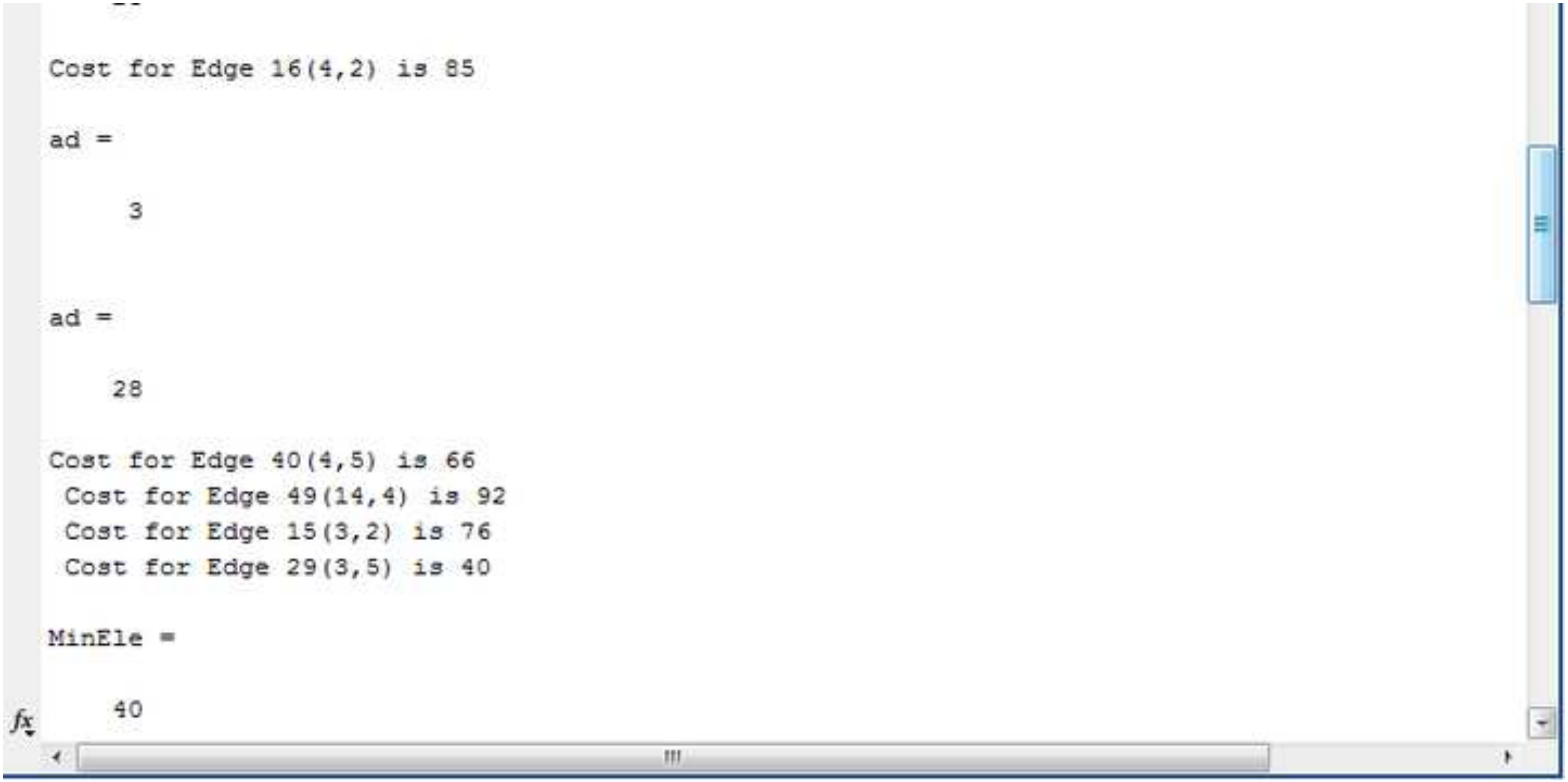
```
here. If you just want an ampersand, the remedy is
simple: Just type `I\&' now. But if some right brace
up above has ended a previous alignment prematurely,
you're probably due for more error messages, and you
might try typing `S' now just to see what is salvageable.

[22] (./template1.aux)

LaTeX Warning: There were multiply-defined labels.

 )
Here is how much of TeX's memory you used:
 2283 strings out of 493027
 33741 string characters out of 6137679
 96887 words of memory out of 5000000
 5808 multiletter control sequences out of 15000+600000
 7650 words of font info for 27 fonts, out of 8000000 for 9000
 1141 hyphenation exceptions out of 8191
 37i,8n,23p,1514b,277s stack positions out of
5000i,500n,10000p,200000b,80000s
<c:/TeXLive/2015/texmf-
dist/fonts/type1/public/amsfonts/cm/cmbx10.pfb><c:/TeX
Live/2015/texmf-
dist/fonts/type1/public/amsfonts/cm/cmbx12.pfb><c:/TeXLive/2015
/texmf-
dist/fonts/type1/public/amsfonts/cm/cmbx7.pfb><c:/TeXLive/2015/texmf-dis
t/fonts/type1/public/amsfonts/cm/cmbx8.pfb><c:/TeXLive/2015/texmf-
dist/fonts/ty
pe1/public/amsfonts/cm/cmex10.pfb><c:/TeXLive/2015/texmf-
dist/fonts/type1/publi
c/amsfonts/cm/cmmi10.pfb><c:/TeXLive/2015/texmf-
dist/fonts/type1/public/amsfont
s/cm/cmmi5.pfb><c:/TeXLive/2015/texmf-
dist/fonts/type1/public/amsfonts/cm/cmmi7
.pfb><c:/TeXLive/2015/texmf-
dist/fonts/type1/public/amsfonts/cm/cmr10.pfb><c:/T
eXLive/2015/texmf-
dist/fonts/type1/public/amsfonts/cm/cmr5.pfb><c:/TeXLive/2015
/texmf-
dist/fonts/type1/public/amsfonts/cm/cmr6.pfb><c:/TeXLive/2015/texmf-dist
/fonts/type1/public/amsfonts/cm/cmr7.pfb><c:/TeXLive/2015/texmf-
dist/fonts/type
1/public/amsfonts/cm/cmr8.pfb><c:/TeXLive/2015/texmf-
dist/fonts/type1/public/am
sfonts/cm/cmsy10.pfb><c:/TeXLive/2015/texmf-
dist/fonts/type1/public/amsfonts/cm
/cmsy7.pfb><c:/TeXLive/2015/texmf-
dist/fonts/type1/public/amsfonts/cm/cmti10.pf
b>
Output written on template1.pdf (22 pages, 1434460 bytes).
PDF statistics:
 158 PDF objects out of 1000 (max. 8388607)
 99 compressed objects within 1 object stream
 0 named destinations out of 1000 (max. 500000)
```

```
76 words of extra memory for PDF output out of 10000 (max. 10000000)
```

attachment to manuscript

Click here to download attachment to manuscript alte.png

Click here to view linked References



```
Cost for Edge 16(4,2) is 85

ad =

     3


ad =

    28

Cost for Edge 40(4,5) is 66
 Cost for Edge 49(14,4) is 92
 Cost for Edge 15(3,2) is 76
 Cost for Edge 29(3,5) is 40

MinEle =

    40
```